Desarrollo Abierto para Internet de las Cosas y Automatización Industrial: Evaluación de Rendimiento en Entornos Críticos

Ricardo Antonio López

Universidad Nacional de la Patagonia

SyA: Sistemas y Automatismos

Trelew, Argentina

lopez.ricardoa@gmail.com

Resumen*— Este trabajo se enfoca en el diseño y desarrollo de un software, con implementación de un prototipo, para aplicaciones en Internet de las Cosas y automatización industrial, que permitió realizar pruebas exhaustivas para evaluar su rendimiento y determinar factores de mérito.*

Se simularon diversos entornos, incluyendo múltiples microcontroladores en redes de diferentes tipos y capas. Se recrearon escenarios con ocurrencia de eventos múltiples en tiempos breves —característicos de entornos eléctricos bajo fallo—, donde el registro preciso es crucial para diagnóstico, detección de puntos de falla y la planificación de inversiones.

*Además, la implementación permitió evaluar el rendimiento de un **RTOS** (Sistema Operativo en Tiempo Real) en modos multitarea y multinúcleo, sobre un microcontrolador de 32 bits, de tamaño moderado, ampliamente compatible y popular en el mercado.*

Durante el desarrollo, se prestó especial atención a una arquitectura modular del software, asegurando escalabilidad e interoperabilidad entre los diversos módulos del sistema.

Palabras clave: *InternetDeLasCosas, Automatización, RTOS, Scada, Sincronización.*

I. Introducción

Para el desarrollo e implementación del concepto propuesto, se escribió un software al que denominaremos **ds7.1,** con tareas definidas y desplegadas en diferentes módulos, para ser portado sobre microcontroladores de amplio espectro, en cuanto a tamaño de procesador(es), memoria y entrada/salida de propósito general **(GPIO)**. Se ha utilizado RTOS y en particular, se lo ha experimentado sobre algunas de las unidades que integraron la red prototipo. De estas unidades, se eligió el microcontrolador ESP32 WROOM de Espressif Systems, por su alta capacidad al poseer dos núcleos procesadores y poder así llevar a cabo un estudio más profundo del problema de sincronización horaria [2][3] y registro de eventos [8], que es en si uno de los subproductos que surgen del presente trabajo.

Sobre el proyecto global, se plantearon desde inicio los siguientes objetivos:

1. Garantizar portabilidad del software, sobre la base de utilización de un soporte Hardware que cumplimente cierto estándar, con el objetivo de reducción de costos y facilidad de su adquisición en el mercado.

2. Utilizar protocolos de comunicaciones sobre diferentes capas físicas y de comunicaciones, todas del estándar de Sistemas Distribuidos [1], con una capa de aplicación [5] adecuada a las necesidades de cada caso a tratar.

3. Lograr escalabilidad de proyectos en un modo simple, mediante la modificación de archivos de configuración, evitando modificaciones en el núcleo del software. Al mismo tiempo, adecuarse a las necesidades de usuario, posibilitando realizar algoritmos específicos de automatismos y de ser necesario agregar comandos de control para cada aplicación [5].

4. Disponer libremente de todo el Software y la documentación involucrados en el proyecto, para posibilitar transferencia de tecnología y eliminación de la cautividad.

La red de la ***Figura 1***, otorga una vista global del conjunto bajo estudio. Esta red se compone de diferentes Dispositivos Electrónicos Inteligentes (IEDs), interconectados mediante diferentes capas físicas y de comunicaciones normalizadas [1][5][9].

Cabe destacar que el software objeto de este proyecto se basa en desarrollos anteriores [4][8]. Estos desarrollos que fueron surgiendo en los últimos años, se han ido implementando en diferentes microcontroladores, con escritura en lenguajes C y en assembler [6][9]. Varias de las rutinas se han reutilizado con ligeras adecuaciones dado que provienen de desarrollos suficientemente probados, con algunos de ellos aún funcionando en empresas eléctricas donde se ha desempeñado en distintos roles, quien esto escribe. El software se fue actualizando, siguiendo la evolución tecnológica del hardware y en este proyecto, se ha migrado íntegramente a C++, en los diferentes Entornos Integrados de Desarrollo **(IDEs)** actuales que poseen los fabricantes.

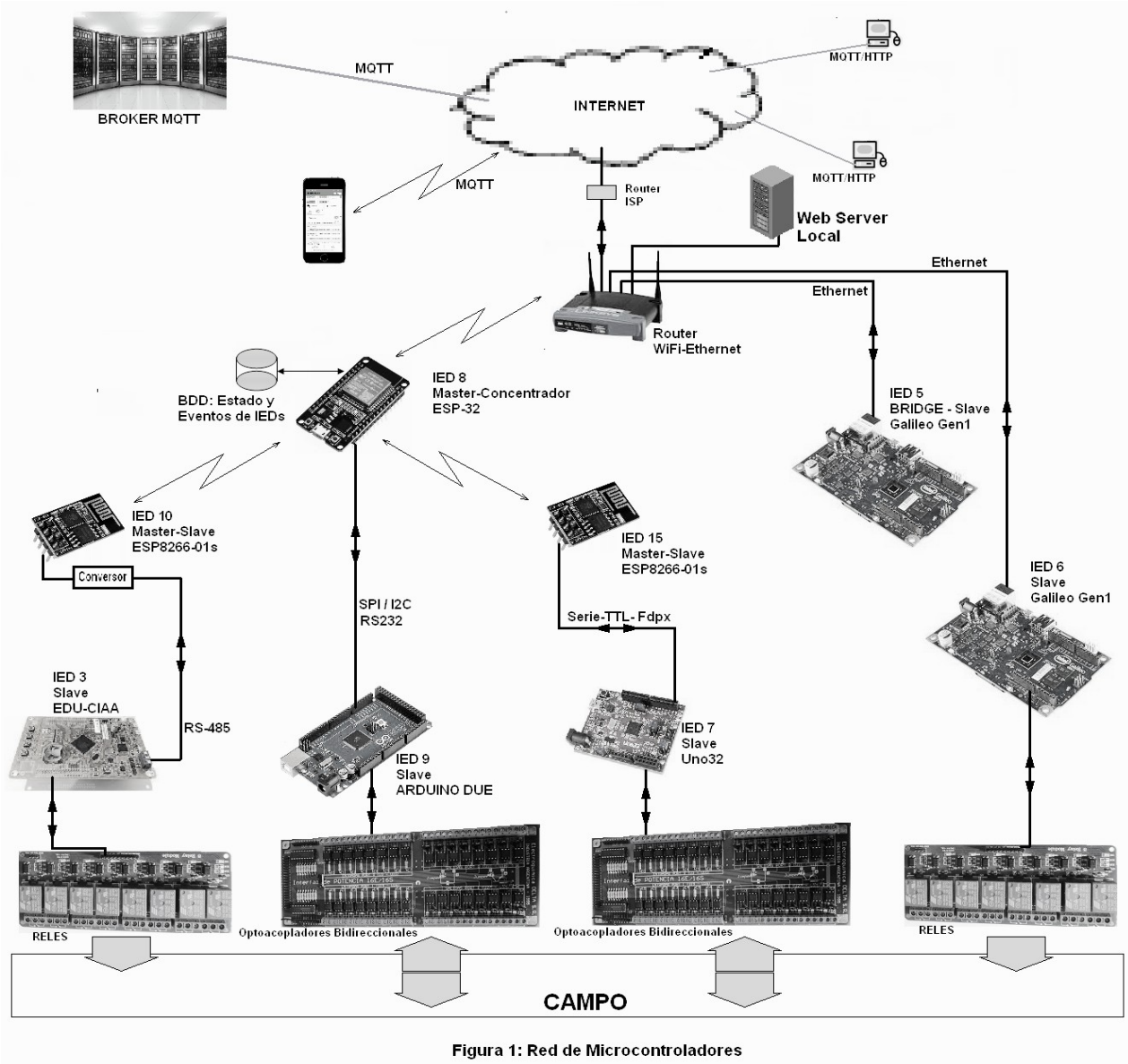

Figura 1: Red de Microcontroladores

Otro de los aspectos que se ha tenido muy en cuenta, es la de hacer posible la integración de IOT en sistemas preexistentes, para reemplazo progresivo de aquellos ya obsoletos. Esta característica puede resultar interesante en ambientes como los de producción agropecuaria, donde la evolución tecnológica suele ser lenta, o instalaciones de Pyme, que puedan poseer una electrónica que aunque obsoleta, aún sigue en funcionamiento por confiabilidad o factores económicos y de oportunidad. Concretamente para este propósito, en el ambiente de simulación se han utilizado diferentes placas de distintos fabricantes, con diferente periferia de I/O y capas físicas y de comunicaciones como Serie FDX, RS485 o Ethernet, además de la ya clásicas por su ubicuidad como WiFi, USB o Bluetooth.

II. RED DE IEDS

Se presenta aquí una descripción de la filosofía conceptual de la Red. Se define la distribución jerárquica de los IEDs, en relación con lo previsto en el software particularmente en sus tablas de configuración, para lograr un funcionamiento coherente y eficiente.

Como se consigna en la Figura 1, cada IED de la red posee un ***Identificador Único (Id)***, para su uso en la capa de aplicación, independiente de su dirección de Red. Los diferentes protocolos de comunicaciones empleados en el sistema han determinado la generación de ***Canales,*** de modo tal que cada canal es tratado con sus particularidades en cada capa. La tabla de la Figura 2 presenta una síntesis de funciones y características de los IEDs de la red bajo estudio.

El núcleo del sistema lo compone un Maestro Concentrador denominado ***CoMaster***, que se encarga de las comunicaciones externas con internet y que periódicamente, a través de la red interna consulta a diferentes IEDs esclavos ***(Slaves)***. Estos, a su vez están en vinculo directo con el campo a través de su GPIO.

El CoMaster, mediante consulta periódica, se nutre de los estados y eventos que se generan en el campo, los incorpora a sus propias tablas de datos, para luego permitir su requerimiento desde Servidores y/o diferentes Interfases Hombre Máquina **(MMI)**, situadas tanto en la red interna o la externa (Internet), característica de un Sistema Distribuido [1][2]. Esto es, el CoMaster contiene las tablas centrales de tiempo real, de trabajo de todo el sistema subyacente y por ende es quien alimenta al Servidor WEB y a las distintas MMIs.

Podrá haber, según fuera el tamaño de la instalación, un Servidor que contenga una BDD central, relacional y más elaborada del sistema, que permita la consulta y hasta la existencia de pantallas con mímicos visuales que representen diferentes áreas para la operación del sistema bajo supervisión y control.

Por su lado, los Slaves se encargan de la adquisición directa desde el campo donde, por estar aliviados de tareas complejas de comunicación debido a la existencia de capas más simples, se desenvuelven con mayor velocidad, libres de procesos con esperas que puedan monopolizar la CPU. Estos IED poseen amplia y variada periferia, son concebidos para dedicarse a los procesos locales y de recepción de eventos, con sincronización horaria interna y precisión por debajo de 1 ms.

#	IED	Función	Hardware	BDD	Automatismo Local	Estado
1	8	Concentrador-Master	ESP32	Global	Ninguno	Desarrollado
2	10	Master - Slave	ESP8266-01	Parcial	Adquisición	Desarrollado
3	15	Master - Slave	ESP8266-01	Parcial	Adquisición	Desarrollado
4	9	Slave	Arduino DUE	Local	Semaforización	Desarrollado
5	7	Slave	UNO 32	Local	Relés Tempo	Desarrollado
6	5	Slave - Bridge	Intel Galileo Gen1	Local	Adquisición	En prueba
7	6	Slave	Intel Galileo Gen1	Local	Relés Tempo	Desarrollado
8	3	Slave	Edu-CIAA	Local	Adquisición	En prueba

Figura 2: Tabla de Funciones de IEDs

Las placas constitutivas de los slaves, típicamente están excluidas del manejo de WIFI y en general también de un Stack TCP/IP – procesos inherentemente “pesados” -, lo que les permite sobrellevar fácilmente rutinas, usualmente con interrupciones para la ejecución de tareas críticas. Normalmente, los procesos del Stack de comunicación externa, se apoderan del procesador en ocasión y con duración impredictivas, haciendo que el tratamiento interruptivo determinístico de los procesadores sea prácticamente inviable. Esto exige la adopción de placas que tengan un manejo interruptivo más predictivo. Los procesadores de la gama estándar, son de 32 bits (54 puertos digitales I/O, 8 PWM, 14 analógicos, 3 puertos serie, SPI, etc.), atributos que los hacen totalmente adecuados para escalabilidad. Ello permite alta velocidad en adquisición de eventos [12], retroalimentación y control automático generado en el propio IED y transferencia de comandos desde una MMI[15], por mencionar algunas de las actividades más críticas en el campo.

Para posibilitar la transferencia de datos en un modo ubicuo, se utiliza comunicación directa en modo WEB o indirecta mediante protocolo MQTT [15][16]. En ambos casos con un protocolo de aplicación propietario de código libre[5][7], denominado Mara-1, en su versión actual v6.1.

Los procesos de comunicación y los de adquisición y control soportados en distintos procesadores, permiten tener registro de eventos con precisión horaria interna por debajo de milisegundo [9][10][11]. En general se utiliza NTP en internet, para obtención de UTC (Tiempo universal coordinado), con ajuste derivativo [2][3], para asegurar una precisión externa del orden de 10 ms.

Para ciertos casos donde la precisión horaria por requerimiento de las aplicaciones tenga una exigencia mayor, se ha previsto el agregado de GPS con Temporización por Flanco, que permite llegar a resoluciones del orden de microsegundos. Ello es particularmente útil por ejemplo, en localización de puntos de falla en líneas de alta tensión, como la línea larga Futaleufú - Puerto Madryn, de 570Kmts. Ello explica - como se verá más adelante en el apartado específico -, la búsqueda de precisiones elevadas en el registro de eventos y sincronización entre IEDs.

Los distintos IEDs simbolizados en funcionamiento jerárquico en la Figura 1, están provistos del mismo software ds7.1. Se varía su configuración para funcionalidad, mediante archivos (*.h) y de esta manera se ha logrado implementar la red descrita, para pruebas de funcionamiento con diferentes placas de distintos fabricantes, de distintos tiempos de aparición en el mercado y consecuentemente diferente tecnología, cantidad de procesadores y memoria, a saber: Uno32 (Microchip - 32 bits), Galileo Gen1 (Intel Pentium - 16bits), Arduino DUE (ARM Cortex-M3 - 32 bits), ESP8266 (Tensilica L106 - 32 bits), EDU CIAA (NXP LPC4337 dual core - 32 bits), ESP32 (Xtensa LX6 dual core - 32 bits). Esta lista no es extensiva dado que pueden agregarse otras placas de fabricantes que cumplan el estándar.

Como ya se ha indicado, las pruebas efectuadas se basaron en el IED más potente de los enunciados, a quien se ha asignado la función más critica que es la de funcionar como CoMaster de la red. Este procesador tiene a su vez su propia GPIO que le permite interactuar con el campo como otro IED de la RED. De hecho, la consulta como IED local está inscrita en la agenda de consulta del proceso CoMaster. Esto es, el CoMaster cumple las funciones independientes de: Encuesta de la red subyacente, encuesta de su propia periferia y consulta desde un nivel superior. Debido a ello, se ha distribuido la tareas en diferentes procesos y en ciertos casos, de ser posible, en diferentes núcleos de un mismo micro-controlador.

Por otro lado, ciertos IED que se encuentran descendiendo jerárquicamente en la red, también tienen las funciones concurrentes de ser: a) Master de su sub-red, b) Adquisidor de datos y controlador de su campo, con su propia GPIO y c) Slave para el CoMaster situado jerárquicamente arriba. Por ejemplo, en la Figura 1, esta triple función la tienen los IEDs de Id: 10 y 15. Las distintas configuraciones y consecuentes roles de los microcontroladores, se logran como se ha dicho, con cambios dentro de los archivos de configuración del software.

Con todo lo descrito hasta aquí, se efectúa un resumen de las siguientes características principales contenidas en el sistema ds7.1.

II.1. CARACTERISTICAS

1. Software sobre Free RTOS Multitask y Multicore. Algunas tareas son del tipo colaborativo con Timers. Permite la realización de las tareas de sistema, comunicaciones y procesos de usuario. El usuario puede programar su propio automatismo y asignarle un slot de ejecución.
2. CPUs: ESP32, ESP8266, NodeMCU, Arduino DUE, MEGA y otros compatibles.
3. Comunicación externa por WIFI. Sincroniza la Hora Global con NTP, con sincronización externa del orden de 10 ms con ajuste derivativo.
4. Sincronización interna con precisión por debajo de 1 ms. Registro de eventos Digitales y Analógicos. Los eventos digitales se clasifican en: ***Rápidos y Lentos***. Los eventos Rápidos se registran por ***interrupción*** del hardware, con una discriminación típica de ± 1 dms (diez-milisegundo = $s/10^{-4}$). Los eventos Lentos se registran por ***polling,*** con una discriminación típica de 10 ms ($s/10^{-2}$). Registro de Eventos por Tipo, Código y Motivo, con TimeStamp.
5. Software/Hardware configurable y escalable, con tablas de parámetros con Registro en EEProm.
6. Comunicación directa e indirecta para adquisición de datos y control (Web Server y PubSub MQTT) sobre el CoMaster, que mantiene Tablas de Datos del sistema Global. MMI por Dashboard, SmartPhone (App) o PC/MAC (Aplicación WEB).
7. Comunicación interna en red mediante diferentes capas físicas estándar: WIFI, Ethernet, Bluetooth, SPI, red cableada serie (Cable físico FDX, o RS485, o Fibra óptica).
8. Entradas y salidas analógicas y digitales, escalables con distintas placas de adquisición.
9. Envío de eventos y recepción de Comandos por Mensajería Instantánea, MQTT o APP customizada.
10. Comandos de Adquisición de datos, Control, Diagnóstico y Configuración en línea, sobre subcapa de aplicación Mara 1-v6.1 (propietario / abierto).

III. SOFTWARE

Todo el concepto que se ha descrito hasta aquí, reúne la elección de arquitectura del Hardware, Comunicaciones y un Software propietario, abierto que se ha publicado en un repositorio de uso libre y dado que uno de sus atributos es su utilización en Domótica, rama de IOT, se lo ha denominado “Domo” Sapiens (ds7.1) y actualmente se encuentra en su versión 7.1.6.

III.1. ESTRUCTURA

Se describe aquí la filosofía conceptual del Software, que se ha escrito en C++ y separado en 18 fuentes (actualmente), que contienen los diferentes módulos para facilitar modificación, escalabilidad y otras necesidades que pudieren surgir. En el desarrollo se ha tenido en cuenta especialmente, la posibilidad de creación de procesos locales ad hoc, a quienes se puede asignado su respectivo “slot” de ejecución dentro de las tareas de RTOS, permitiendo la confección de los algoritmos necesarios para el proceso local del cual se encarga el micro-controlador, ello independientemente de sus tareas globales predeterminadas, como son la adquisición del estado y comunicación con el CoMaster.

Las tareas se han distribuido en diferentes Tareas del RTOS, en diferentes núcleos cuando existe esa posibilidad, asignando al Núcleo 0 el manejo de WIFI y el stack TCP/IP donde requieren sus propios tiempos y oportunidad de ejecución, algunos de ellos críticos. Se reserva el Núcleo 1 para las interrupciones del usuario y el manejo del Clock Interno con ajuste derivativo, que también requiere tiempos precisos de ejecución para mantener la granularidad y precisión que se han impuesto como principios preliminares de diseño.

Se ha dotado al software de una salida de debug que se otorga en la ***FIGURA 3***. donde se puede apreciar el manejo de mensajes con los IEDs, ya sea en el Rol de Master, Slave o Master-Slave. También puede observarse la ocurrencia de eventos de distinto tipo 0 al 3, especialmente.

El protocolo de Aplicación mara-1 se ha elegido en principio, por el elevado conocimiento que se tiene del mismo dado que fue objeto de proyectos específicos. Al ser abierto, posee un forma muy sencilla de crear nuevos

comandos adecuados a una aplicación específica, por encima de los generales de su inicio. Su formato es de un byte con la secuencia comando y argumentos: comX, arg1, arg2, arg3…Donde X= Nro. de Comando (de 0 a 254) y argX son los argumentos que se adicionen para el mismo. Algunos de los comandos generales que se han definido son: Requerimiento de Estado del IED, Requerimiento de Eventos, Puesta en Hora, Lectura de Variables para Debug, por listar los más importantes.

Cabe destacar que el protocolo posee, ademas de su capa de aplicación, dos capas de comunicaciones (Física y Transporte) donde se verifican CheckSum, secuencia de paquete, direcciones de origen y destino y bytes de encuadre. Este formato de tres capas, se utiliza especialmente en comunicación serie FDX y multipunto. En situaciones en que se requiere el estándar [1], los paquetes del protocolo son encapsulados en TCP/IP.

III.2. Tiempo y Sincronización

En la Unidad CoMaster, el UTC se recibe por NTP. Internamente, se define una variable **PeHStatus,** que indica el modo en que el UTC es aplicado a la CPU local. Al inicio, con PeHStatus que va en aumento desde 0 hasta 2, el UTC se aplica directamente al reloj local en tanto se pueda determinar el tiempo más probable.

Consecuentemente, esta imposición del valor de UTC al reloj local hace que el mismo pueda ir hacia atrás, rompiéndose el principio de monotonicidad necesario para el correcto registro de eventos secuenciales. Pero ello solo sucede al arranque del microcontrolador.

En cada ajuste de tiempo, denominado **PeH** (puesta en hora), se calcula el **Sesgo** existente entre la hora local y la de llegada del paquete NTP, corregido por Algoritmo de Cristian [3]. Si el Sesgo es positivo, por convención propia, se indica que el microcontrolador adelanta respecto a la hora UTC. En el arranque, que es la situación que se ha presentado en el volcado central de la Figura 3, se observa que se efectúan lecturas consecutivas, con diferencias del orden de 15 segundos entre ellas, hasta que se toma un UTC que ocasiona un Sesgo bajo (de 23 dms, en este caso). En ese instante es cuando concluye la PeH por imposición. Se pone la variable PeHStatus = 3 y a partir de allí, el UTC que se obtenga se aplica sólo corrigiendo la deriva, de modo que el reloj local nunca vuelve hacia atrás conservando así su monotonía.

Lo antedicho implica que se garantiza que la corrección, que siempre será igual a la granularidad temporal, en este caso de ± 1 dms y se hace con un período de aplicación variable e inversamente proporcional al Sesgo resultante en la lectura, con el objetivo de acelerar la convergencia a la hora UTC.

```
Hola!. Comencemos... uCNet4: Domo Sapiens1 - versión: 7.1.4. 22/jul/2024. 13:11  IED: 05. Esp32
Causa de Anterior Reset: 0.  Modo Reset: 0
Previsión de apagado de Outputs... Esperando si se pulsa Boton2 para cargar la EEPROM por Default..
Lectura de Parámetros desde EEPROM ...EEPROM successfully committed
Access Point: LaVidaEsBella
Servidor MQTT: test.mosquitto.org
INICIALIZANDO I/O...I/O Inicializada.
Connecting to LaVidaEsBella .....WiFi connected . IP address: 192.168.1.135
. El Tiempo local es: 2024-6-12   17:30:12.0000   DoW: 0   DoY: 180   PeHStatus: 0
Fin del SETUP. Ahora...Loop!..... Tareas Creadas.....

Trama Publicada: uCNet4Mqtt/Ans/05/91 01 20 20 04
Trama Publicada: uCNet4Mqtt/Ans/05/.-8 Ev tipo: 3 IED: 05. IED Up.EEProm SAVE.! . Reset:0  TimeStp: 2024/8/21  8:27:19.9
Trama Publicada: uCNet4Mqtt/Ans/05/.-7 Ev tipo: 3 IED: 05. PeH. Sesgo: 0. PeHStatus: 1       TimeStp: 2024/8/21  8:27:19.0
Trama Publicada: uCNet4Mqtt/Ans/05/.-6 Ev tipo: 3 IED: 05. PeH. Sesgo: 0. PeHStatus: 2       TimeStp: 2024/8/21  8:27:32.0
Trama Publicada: uCNet4Mqtt/Ans/05/.-5 Ev tipo: 3 IED: 05. PeH. Sesgo: -5847. PeHStatus: 3  TimeStp: 2024/8/21  8:27:47.59689
Trama Publicada: uCNet4Mqtt/Ans/05/.-4 Ev tipo: 3 IED: 05. PeH. Sesgo: -1290. PeHStatus: 3  TimeStp: 2024/8/21  8:28:2.64246
Trama Publicada: uCNet4Mqtt/Ans/05/.-3 Ev tipo: 3 IED: 05. PeH. Sesgo: -204. PeHStatus: 3   TimeStp: 2024/8/21  8:28:17.65332
Trama Publicada: uCNet4Mqtt/Ans/05/.-2 Ev tipo: 3 IED: 05. PeH. Sesgo: 23. PeHStatus: 3     TimeStp: 2024/8/21  8:28:32.23
Trama Publicada: uCNet4Mqtt/Ans/05/.-1 Ev tipo: 3 IED: 05. PERDIDA respuesta IED: 3357      TimeStp: 2024/8/21  8:28:36.3357
CountLoop00: 0. CountLoop10: 3994. CountLoop11: 766. CountLoop12: 799. Millis: 1670327
Temperatura *C *F:  22.0   71.5  Temperatura PROM *C *F: 21.7  71.1
....
 Emision NTP packet...Leyendo Pack NTP. Delay[ms]: 20.60.100.140.180.
  Hora NTP. Packet recibido= 48. Unix time= 1724015077. delayLectura: 2200 dms [ms/10]
  El UTC corregido alg. Cristian es: 2024-8-18  18:04:38.0243 .Fracción en décimas de ms [dms]   PeH: (Com 12).
ComError: 0x10. Dif.dias: -66. Dif.Horaria[s]: -2035. fraccionDMS[dms]: 5. Sesgo[dms]: 0. PeriNTP: 15.
(local...) TimeStp[i]:  0x18  8  12  12  4  26  F3
(entrante) BufApli[i]: 0x18  8  12  12  4  26  F3
. El Tiempo local es: 2024-8-18   18:04:38.0243   DoW: 7   DoY: 231   PeHStatus: 1
...
** PeH-FL: FE 11 FF 40 02 12 18 08 12 12 09 38 88 13 07 3E 37 – Time:  2024-8-18   18:09:56.5000
```

Figura 3: Dump del Debug de arranque, puesta en hora inicial y emisión de PeH del IED.

Para lograr el objetivo de ajuste derivativo, se corrije el valor del incremento que se aplica al reloj a efectos que su velocidad varíe, en cada ciclo del Timer Hardware del microcontrolador. Por ejemplo, si el Sesgo obtenido es positivo, señal que el reloj local adelanta, se decrementa en 1 dms (un *Leap* dms), el valor que sistemáticamente se aplica al reloj y consecuentemente su indicación atrasará, intentando converger a un Sesgo teórico nulo en la próxima obtención del UTC. Normalmente, de un modo aleatorio de múltiples causas, la llegada de la nueva lectura traerá aparejado un nuevo Sesgo y su signo y módulo, determinará un nuevo cálculo del ajuste derivativo a aplicar al nuevo ciclo y así sucesivamente.

Si el ajuste derivativo no lograra su objetivo de convergencia, ante la existencia de un reloj deficiente que acumule un corrimiento creciente entre las sucesivas lecturas de NTP, se podrán hacer lecturas más frecuentes para mejorar el ajuste derivativo. Si este procedimiento no diera resultado, se podrá recurrir a otro método de obtención del UTC, como puede ser el agregado al hardware de un GPS con Emisión de Flanco **pps** (pulse per second), como se verá luego en el ensayo que se ha planteado.

En conclusión, con lo expuesto en este apartado, se logran buenas aproximaciones respecto al UTC, para la sincronización externa del orden de 10ms, con lectura desde internet con NTP y de pocos microsegundos con GPS con Emisión de Flanco. El valor de tiempo obtenido en el CoMaster, se puede emitir por broadcast con correcciones, a los otros IEDs de la red interna [3][10].

IV. Escenario de Prueba

IV.1. Consignas preliminares

Se describen aquí las condiciones del ensayo al que se sometió a un IED, para evaluación de desempeño en eventos digitales. Como se ha consignado anteriormente en Características (ítem 4), el software puede registrar Eventos Rápidos y Lentos. La situación más crítica que es la de registro de eventos rápidos, se presenta a menudo dentro de los problemas de ingeniería a resolver con microcontroladores. Esta problemática particular, se encuentra en campos diversos como Control, Robótica, Mediciones, etc. Se describe a continuación un escenario posible de medición en el campo de la Ingeniería eléctrica, que sirve como marco de referencia para la experimentación efectuada.

Como es conocido, el disturbio que se genera ante una falla eléctrica, viaja a la velocidad de propagación electromagnética en el medio -en este caso el cable empleado para la conducción de la energía eléctrica en la línea -, llegando dicho frente en diferentes tiempos a ambos extremos de la misma. La detección y medición del tiempo de llegada de ambos eventos, en forma independiente por IEDs situados en cada extremo y sincronizados por UTC entre si, con precisión externa de pocos microsegundos, permite definir, con errores menores a un kilómetro, la posición del punto de falla en la línea. Ello redunda en buena velocidad de reparación y reposición del servicio, con disminución del costo de falla, al disminuir el monto de las multas y demoras generalmente causantes de dificultades diversas. Esta es una síntesis muy simplificada de un desafío donde se requiere elevada precisión de registro temporal. Sobre este punto se volverá en las conclusiones, como un interesante desafío de resolución con IEDs en red.

Se eligió este escenario dado que los ambientes eléctricos - especialmente los de generación, transporte y distribución-, son de alta exigencia por el tipo, cantidad y criticidad de los eventos que ocurren en modo aleatorio. A la vez son relativamente simulables y predecibles – obviamente, no en cuanto al momento de ocurrencia -, como medio de prueba para desarrollos. La normativa emitida por el ente regulador argentino (ENRE), tiene a su vez varias exigencias, que se traducen en la necesidad de precisión en los desarrollos para cumplir con las exigencias del mercado y de la compañía Administradora del Sistema Interconectado Nacional (SIN). Esa exigencias se traducen por ejemplo, en la necesitad de registrar eventos con precisión de milisegundo, determinando que la sincronización de los distintos procesadores que conforman un gran sistema distribuido, también tengan una sincronización externa del mismo orden y que los eventos, lleguen a la compañía administradora con una antigüedad no mayor a 10 segundos, por medios de comunicaciones redundantes, de distintas tecnologías y. El objetivo central de tales requerimientos es el de contar con

información global, fidedigna del sistema para determinar causas y efectos en los caso de falla y determinar económicamente al causante de la falla para la aplicación de severas multas. Por un lado, se efectúan diagnósticos que permiten encontrar vulnerabilidades de la red y de esa forma destinar la inversión. Por otro lado, se produce un sistema de premios y castigos para los actores, componiendo así un incentivo económico para fortalecer las redes y mejorar los sistemas de protección de las mismas. Este ambiente de profusas variables es un excelente medio de prueba para desarrollos y la envergadura de los mismos puede ir de uno a múltiples procesadores interconectados en un sistema distribuido [1]. En esta situación son necesarias distintas capas físicas para interconexión, distintos protocolos [1][2] para implementar TCP/IP que es casi en su totalidad el Stack utilizado con sus diversos servicios (SNTP, HTTPS, MQTT, etc) y múltiples aplicaciones para distintos objetivos (medición comercial y de operación, estados, eventos, control de interruptores y seccionadores, soporte de protecciones , etc.). Quien esto escribe se ha desempeñado profesionalmente durante muchos años como responsable del servicio.

IV.2. Escenario - Alcance

Para la prueba del sistema descrito hasta aquí, se ajustó el alcance de la experimentación estableciendo las siguientes consignas:

- Se cablearon las salidas de un IED ESP32 con Id:01, en forma directa (back to back), hacia sus homónimas de entrada en otro ESP32 con Id: 05 como se aprecia en la **Figura 4a** y en el circuito de la **Figura 4b**.

- El IED 01 sincroniza su reloj por NTP desde los servidores públicos en internet. Se espera que en esta condición pueda llegar a existir un Sesgo Temporal con el UTC, que ronda en 10 ms. De cualquier manera, ello no es importante porque queda fuera del alcance de este estudio y no afecta la precisión que puede lograrse en la red interna.

- En el mismo procesador, se implementó una rutina de Sincronización de hora, emulando un GPS con Emisón de Flanco pps. El IED se toma como estrato de referencia de la hora actual sincronizada por NTP y el dato se emite por puerto serie 500 ms antes del inicio de cada segundo. La emisión del paquete de PeH puede observarse en la última fila del Dump presentado en la Figura 3, fila que se repite:

**** PeH-FL: FE 11 FF 40 02 12 18 08 12 12 09 38 88 13 07 3E 37 – Time: 2024-8-18 18:09:56.5000**

Obsérvese que la hora se emite por timer interno en 5000 dms desde inicio del segundo anterior. En el pasaje por cero del segundo asociado siguiente, se emite un flanco digital. Estas señales son ingresadas al IED Id:05 por las puertas adecuadas (Rx1 y DI), para ser aplicadas a la rutina de PeH. De esta forma el IED 05 toma como estrato superior al IED 01 y sincroniza su hora con él, con una precisión del orden de microsegundos, necesaria para el alcance dado a esta experimentación.

- En el IED 01, también se ha programado una rutina que permite generar ráfagas de eventos. En este caso, mediante un comando (com17) por la app en MQTT, de generación de eventos. Implementado ad hoc, permite generar ráfagas sobre 16 puertos de salida distintos, elegidos del grupo GPIO del IED 01.

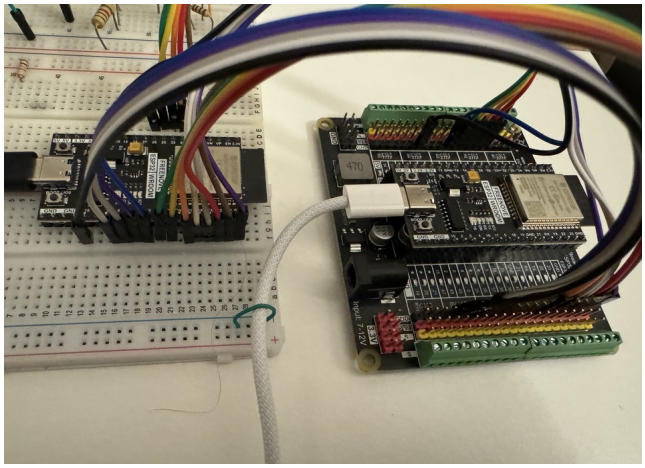

Figura 4a: Escenario de estudio.

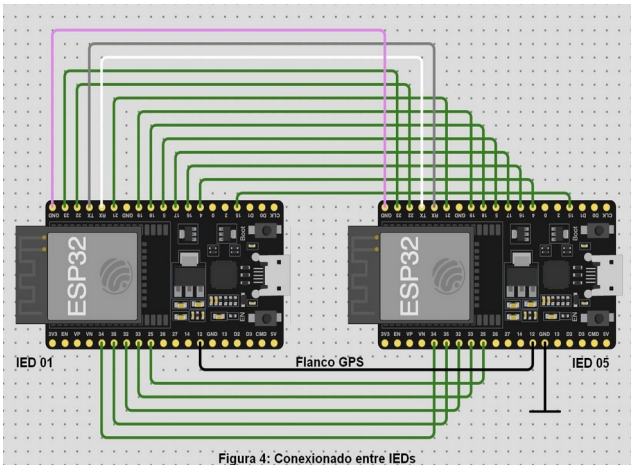

Figura 4b: Conexionado entre IEDs.

- La correspondencia entre Nro. de evento y Nro. de GPIO y además la **dirección indirecta**, de Puerto y Número de Bit, se indica en detalle en la tabla de la **Figura 5b,** columnas 1 a 4 para el IED 01 y columnas 1, 2, 7 y 8 para el IED 05.

- Los eventos de la ráfaga se emiten en un modo cuasi simultáneo, en un ciclo FOR del software, de modo que la diferencia temporal de emisión entre si para cualquier evento, está por debajo de 1 microsegundo, por lo que a los efectos de este ensayo se los considera simultáneos. Luego de emitida la primera ráfaga de 16 eventos, se hace una pausa de 2 ms y se la repite nuevamente para luego detener la emisión. O sea, se pueden emitir en cualquier momento, dos ráfagas consecutivas de 16 eventos distintos cada una, con una diferencia temporal entre ellas de 2 ms.

- En el mismo IED 01, estas direcciones se han declarado en la tabla de configuración del software para Eventos Rápidos, con lo cual los eventos que se emitan, se registrarán por loop interno también como eventos de entrada en el propio Bufer de Eventos del IED 01. Ello se hace para tener precisión del TimeStamp de emisión, de cada uno de un total de 32 eventos (2 repeticiones de 16), numerados en la columna 1 de la Tabla de Figura 5b. Estas ráfagas emitidas por el IED 01, ingresan a los puertos de entrada homónimos de la GPIO del IED 05.

```
Message arrived: uCNet4Mqtt/Com/05/17.20.
COMANDO default  BufApli: 17 20
Resp Comando MQTT. Canal: 3. CBufApli[Canal]: 3
*** Eventos. PEvIngreso:   8. PEvOut: 0. BufEventos: 05 80 18 08 12 11 2E 0A 38 0C
*** Eventos. PEvIngreso:   9. PEvOut: 0. BufEventos: 05 81 18 08 12 11 2E 0A 38 0C
*** Eventos. PEvIngreso:  10. PEvOut: 0. BufEventos: 05 82 18 08 12 11 2E 0A 39 0C
*** Eventos. PEvIngreso:  11. PEvOut: 0. BufEventos: 05 83 18 08 12 11 2E 0A 38 0C
*** Eventos. PEvIngreso:  12. PEvOut: 0. BufEventos: 05 84 18 08 12 11 2E 0A 39 0C
*** Eventos. PEvIngreso:  13. PEvOut: 0. BufEventos: 05 85 18 08 12 11 2E 0A 39 0C
*** Eventos. PEvIngreso:  14. PEvOut: 0. BufEventos: 05 86 18 08 12 11 2E 0A 38 0C
*** Eventos. PEvIngreso:  15. PEvOut: 0. BufEventos: 05 87 18 08 12 11 2E 0A 39 0C
*** Eventos. PEvIngreso:  16. PEvOut: 0. BufEventos: 05 88 18 08 12 11 2E 0A 38 0C
*** Eventos. PEvIngreso:  17. PEvOut: 0. BufEventos: 05 89 18 08 12 11 2E 0A 38 0C
*** Eventos. PEvIngreso:  18. PEvOut: 0. BufEventos: 05 8A 18 08 12 11 2E 0A 39 0C
*** Eventos. PEvIngreso:  19. PEvOut: 0. BufEventos: 05 8B 18 08 12 11 2E 0A 39 0C
*** Eventos. PEvIngreso:  20. PEvOut: 0. BufEventos: 05 8C 18 08 12 11 2E 0A 38 0C
*** Eventos. PEvIngreso:  21. PEvOut: 0. BufEventos: 05 90 18 08 12 11 2E 0A 80 0C
*** Eventos. PEvIngreso:  22. PEvOut: 0. BufEventos: 05 91 18 08 12 11 2E 0A 80 0C
*** Eventos. PEvIngreso:  23. PEvOut: 0. BufEventos: 05 92 18 08 12 11 2E 0A 80 0C
*** Eventos. PEvIngreso:  24. PEvOut: 0. BufEventos: 05 00 18 08 12 11 2E 0A 4C 0C
*** Eventos. PEvIngreso:  25. PEvOut: 0. BufEventos: 05 01 18 08 12 11 2E 0A 4C 0C
*** Eventos. PEvIngreso:  26. PEvOut: 0. BufEventos: 05 02 18 08 12 11 2E 0A 4D 0C
*** Eventos. PEvIngreso:  27. PEvOut: 0. BufEventos: 05 03 18 08 12 11 2E 0A 4C 0C
*** Eventos. PEvIngreso:  28. PEvOut: 0. BufEventos: 05 04 18 08 12 11 2E 0A 4D 0C
*** Eventos. PEvIngreso:  29. PEvOut: 0. BufEventos: 05 05 18 08 12 11 2E 0A 4D 0C
*** Eventos. PEvIngreso:  30. PEvOut: 0. BufEventos: 05 06 18 08 12 11 2E 0A 4C 0C
*** Eventos. PEvIngreso:  31. PEvOut: 0. BufEventos: 05 07 18 08 12 11 2E 0A 4D 0C
*** Eventos. PEvIngreso:  32. PEvOut: 0. BufEventos: 05 08 18 08 12 11 2E 0A 4C 0C
*** Eventos. PEvIngreso:  33. PEvOut: 0. BufEventos: 05 09 18 08 12 11 2E 0A 4C 0C
*** Eventos. PEvIngreso:  34. PEvOut: 0. BufEventos: 05 0A 18 08 12 11 2E 0A 4D 0C
*** Eventos. PEvIngreso:  35. PEvOut: 0. BufEventos: 05 0B 18 08 12 11 2E 0A 4D 0C
*** Eventos. PEvIngreso:  36. PEvOut: 0. BufEventos: 05 0C 18 08 12 11 2E 0A 4C 0C
*** Eventos. PEvIngreso:  37. PEvOut: 0. BufEventos: 05 10 18 08 12 11 2E 0A 80 0C
*** Eventos. PEvIngreso:  38. PEvOut: 0. BufEventos: 05 11 18 08 12 11 2E 0A 80 0C
*** Eventos. PEvIngreso:  39. PEvOut: 0. BufEventos: 05 12 18 08 12 11 2E 0A 80 0C
```

Figura 5a: IED 05. Segmento del Bufer de eventos.

- En el IED 05 se registrarán normalmente los eventos digitales de entrada y se almacenarán en su Bufer. Hay un detalle a observar en las columna 7 y 8 del IED 05 donde se ha declarado: Las primeras 13 posiciones correspondientes al el Puerto 0 de **eventos rápidos** y las 3 restantes se corresponden con las 3 primeras posiciones del Puerto 1 de **eventos lentos**. Esto se hizo así para tener un elemento de juicio en la consistencia en la información recabada, como se verá luego.

IV.3. Análisis de Resultados

El ensayo se efectuó varias veces, se ajustaron errores de implementación y cuando se obtuvieron ensayos consistentes, se registraron listados de los cuales se tomó uno de ellos, que es el volcado en la tabla de la **Figura 5a.** Ambos microcontroladores cuentan con el software ds7.1 y los resultados se clarifican en correspondencia con los Nros de evento en la planilla de la Figura 5b, que pasamos a analizar.

1. La columna 1 otorga en número del evento secuencial.
2. La columna 2 entrega la dirección física de la GPIO, por comodidad fue la misma utilizada en ambos procesadores.
3. Las Columnas 3, 4, 7 y 8 presentan la dirección indirecta de los puertos de 16 bits (**Puerto** y **Nro. de Bit** respectivamente) de ambos procesadores. El puerto 0 es de eventos rápidos, el puerto 1 es de eventos lentos.
4. La columna 5 (en Hexadecimal) otorga datos de identificación del evento de **salida** del **IED 01** (2 primeros bytes) y su TimeStamp (8 bytes restantes).
5. La columna 6 (en Hexadecimal) otorga datos de identificación del evento de **entrada** del **IED 05** (2 primeros bytes) y su TimeStamp (8 bytes restantes).
6. Cada registro de eventos de las columnas 5 y 6, posee 10 bytes, donde:
    1. Byte1: otorga TIPO (0 para eventos de campo) y dirección del IED (01 del emisor de eventos y 05 del receptor).
    2. Byte 2: Otorga el estado (MSB, 1 bit), el puerto (3 bits inferiores del nibble High) y el NroBit (nibble Low).
    3. Bytes 3, 4 y 5: YY MM DD (año, mes y dia).

4. Bytes 6, 7 y 8: HH mm SS (hora, minuto, segundo).
5. Bytes 9 y 10: Conforman un WORD en formato Little Endian. Contiene fracción de segundo, en dms ($s/10^{-4}$).

7. Para interpretar un evento tomamos como ejemplo el **evento 13**, de la **columna 5** (en negrita): ***05 8C 18 08 12 11 2E 0A 38 0C.*** La capa de Aplicación del protocolo 1-v6.1, tiene la siguiente semántica por cada byte:
   1. 05 - **TIPO**: 0 (0 para eventos de campo) y **Id** del IED: 5 (del receptor de eventos).
   2. 8C - **ESTADO**: 1 (MSB, 1 bit), **PUERTO**: 0 (3 bits inferiores del nibble High) y el **NroBit:** 12 (nibble Low).
   3. **18 08 12** (Bytes 3, 4 y 5) - **YY MM DD** (año, mes y dia): 24/08/18.
   4. **11 2E 0A** (Bytes 6, 7 y 8) - **HH mm SS** (hora, minuto, segundo): 16:46:10.
   5. **38 0C** (Bytes 9 y 10) - Conforman un WORD en formato Little Endian que contiene la fracción de segundo, en dms ($s/10^{-4}$): 0x0C38 = 3128. Luego, el **TimeStamp** del evento de la línea es**:** 2024/08/18 17:46:10.3128

8. Los eventos emitidos por el IED 01, se sitúan en estado 1 para los 16 pertenecientes a la primera ráfaga y en estado 0 para los 16 siguientes, completando un total de 32.

IV.4. CONSIDERACIONES

De la comparativa de los eventos registrados surge lo siguiente:

A) El TimeStamp de la segunda ráfaga es mayor en 20 dms que lo de la primera. Ello era esperable dado que en los ensayos de Puesta en Hora y ajustes de la temporización, donde se había utilizado un Timer Hardware, la hora era muy precisa y con la monotonía esperable. Ello, salvando las diferencias aleatorias que pueden existir respecto a un patrón de hora superior, debido al *drift* del cristal del procesador local, por sus condiciones de temperatura y tensión de alimentación en funcionamiento.

B) Los eventos recibidos por el IED 05 tienen particularidades que se destacan:
   1. El TimeStamp de cada evento recibido es igual al emitido, en casi todos los casos. Esto también es esperable

		UC 01			UC 05			
1	2	3	4	5	6	7	8	
Ev	GPIO	Puerto	NroBit	Uc 01 (Output)	Uc 05 (Input)	Puerto	NroBit	Color
1	**26**	0	0	01 80 18 08 12 11 2E 0A 38 0C	05 80 18 08 12 11 2E 0A 38 0C	0	0	AZ
2	**23**	0	1	01 81 18 08 12 11 2E 0A 38 0C	05 81 18 08 12 11 2E 0A 38 0C	0	1	MA
3	**22**	0	2	01 82 18 08 12 11 2E 0A 38 0C	05 82 18 08 12 11 2E 0A 39 0C	0	2	RO
4	**21**	0	3	01 83 18 08 12 11 2E 0A 38 0C	05 83 18 08 12 11 2E 0A 38 0C	0	3	NA
5	**19**	0	4	01 84 18 08 12 11 2E 0A 38 0C	05 84 18 08 12 11 2E 0A 39 0C	0	4	AM
6	**18**	0	5	01 85 18 08 12 11 2E 0A 38 0C	05 85 18 08 12 11 2E 0A 39 0C	0	5	VE
7	**5**	0	6	01 86 18 08 12 11 2E 0A 38 0C	05 86 18 08 12 11 2E 0A 38 0C	0	6	AZ
8	**17**	0	7	01 87 18 08 12 11 2E 0A 38 0C	05 87 18 08 12 11 2E 0A 39 0C	0	7	VI
9	**16**	0	8	01 88 18 08 12 11 2E 0A 38 0C	05 88 18 08 12 11 2E 0A 38 0C	0	8	GR
10	**4**	0	9	01 89 18 08 12 11 2E 0A 38 0C	05 89 18 08 12 11 2E 0A 38 0C	0	9	BL
11	**15**	0	A	01 8A 18 08 12 11 2E 0A 38 0C	05 8A 18 08 12 11 2E 0A 39 0C	0	A	NE
12	**34**	0	B	01 8B 18 08 12 11 2E 0A 38 0C	05 8B 18 08 12 11 2E 0A 39 0C	0	B	MA
13	**35**	0	C	01 8C 18 08 12 11 2E 0A 38 0C	**05 8C 18 08 12 11 2E 0A 38 0C**	0	C	RO
14	**32**	0	D	01 90 18 08 12 11 2E 0A 38 0C	05 90 18 08 12 11 2E 0A 80 0C	1	0	NA
15	**33**	0	E	01 91 18 08 12 11 2E 0A 38 0C	05 91 18 08 12 11 2E 0A 80 0C	1	1	AM
16	**25**	0	F	01 92 18 08 12 11 2E 0A 38 0C	05 92 18 08 12 11 2E 0A 80 0C	1	2	VE
17	-	0	0	01 00 18 08 12 11 2E 0A 4C 0C	05 00 18 08 12 11 2E 0A 4C 0C	0	0	-
18	-	0	1	01 01 18 08 12 11 2E 0A 4C 0C	05 01 18 08 12 11 2E 0A 4C 0C	0	1	-
19	-	0	2	01 02 18 08 12 11 2E 0A 4C 0C	05 02 18 08 12 11 2E 0A 4D 0C	0	2	-
20	-	0	3	01 03 18 08 12 11 2E 0A 4C 0C	05 03 18 08 12 11 2E 0A 4C 0C	0	3	-
21	-	0	4	01 04 18 08 12 11 2E 0A 4C 0C	05 04 18 08 12 11 2E 0A 4D 0C	0	4	-
22	-	0	5	01 05 18 08 12 11 2E 0A 4C 0C	05 05 18 08 12 11 2E 0A 4D 0C	0	5	-
23	-	0	6	01 06 18 08 12 11 2E 0A 4C 0C	05 06 18 08 12 11 2E 0A 4C 0C	0	6	-
24	-	0	7	01 07 18 08 12 11 2E 0A 4C 0C	05 07 18 08 12 11 2E 0A 4D 0C	0	7	-
25	-	0	8	01 08 18 08 12 11 2E 0A 4C 0C	05 08 18 08 12 11 2E 0A 4C 0C	0	8	-
26	-	0	9	01 09 18 08 12 11 2E 0A 4C 0C	05 09 18 08 12 11 2E 0A 4C 0C	0	9	-
27	-	0	A	01 0A 18 08 12 11 2E 0A 4C 0C	05 0A 18 08 12 11 2E 0A 4D 0C	0	A	-
28	-	0	B	01 0B 18 08 12 11 2E 0A 4C 0C	05 0B 18 08 12 11 2E 0A 4D 0C	0	B	-
29	-	0	C	01 0C 18 08 12 11 2E 0A 4C 0C	05 0C 18 08 12 11 2E 0A 4C 0C	0	C	-
30	-	0	D	01 10 18 08 12 11 2E 0A 4C 0C	05 10 18 08 12 11 2E 0A 80 0C	1	0	-
31	-	0	E	01 11 18 08 12 11 2E 0A 4C 0C	05 11 18 08 12 11 2E 0A 80 0C	1	1	-
32	-	0	F	01 12 18 08 12 11 2E 0A 4C 0C	05 12 18 08 12 11 2E 0A 80 0C	1	2	-
-	**1**	TX	-			RX	-	BL
-	**3**	RX	-			TX	-	GR
	12	**FLout**				**FLin**		NE
-	-	GND	-			GND	-	VI

del Registro de eventos

Figura 5b: Tabla de el análisis

por la elevada precisión que posee el GPS de Emisión de flanco que permite sincronizar ambos procesadores con precisiones de pocos microsegundos.

2. Algunos Eventos Rápidos difieren en 1 dms en exceso, de otros que han sido coincidentes en el instante de emisión. La dispersión no es importante pero es posible imputar el corrimiento de estos eventos a una demora en la gestión de las interrupciones que efectuá el RTOS. Dado que la rutina de servicio de interrupción (ISR) es la misma para todos los eventos registrados, variando sólo el parámetro de su dirección hardware en el llamado y aunque en tiempo de ejecución, la rutina puede hacerse residir en la Internal Ram mediante configuración, para mayor velocidad de ejecución, al SO no le queda más alternativa que las distintas instancias sean ejecutadas en secuencia por cada evento, por aquellos procesadores que estén disponibles aleatoriamente para tal ejecución. Este encolamiento en la ejecución puede ser causa de diferimiento en los tiempos de registro.

3. Las elevadas diferencias que se encuentran en los registros de eventos Lentos 14, 15 y 16 de la primera ráfaga y 30, 31 y 32 de la segunda ráfaga, respecto a su tiempo de emisión y que además tienen el mismo TimeStamp en el registro, siendo que pertenecen a ráfagas distintas, se explica porque aquellos se han configurado deliberadamente para su detección por Polling. Ello se efectúa dentro de la rutina de ejecución sistemática de Toma de Estado, cada 10 ms. Esto se ha determinado así para verificar consistencia en la distinta modalidad de registro entre eventos Lentos y Rápidos. Debido a que la planificación los timers por software dispuestos en la aplicación como método de temporización de tareas, determinan que la rutina de Toma de Estado se ejecute aproximádamente en el instante que los milisegundos son múltiplos de 10 y en cero, se ha dispuesto que el proceso trunque los dígitos decimales por debajo de 10 ms. Esto es así puesto que la granularidad está determinada por el período de la Toma de Estado y la presencia de dígitos decimales por debajo de 10 ms darían una idea equivocada de la precisión. Se observa además que el tiempo de registro de esos eventos, resulta con un TimeStamp en exceso y es igual para todos los eventos Lentos captados en el mismo ciclo. Ello es esperable porque son registrados simultáneamente. En efecto, aunque hay eventos que pertenecen a distintas ráfagas, llegando al IED 05 en diferente tiempo, ellos son recogidos con demora, en el mismo ciclo de toma de estado y se les impone el idéntico TimeStamp.

V. Conclusiones

El desempeño de RTOS fue altamente satisfactorio. El desglose de las tareas con tiempos de ejecución definidos, como las interrupciones asociadas a la rutina del Reloj local y la de ingreso de eventos, se ejecutaron con sorprendente precisión, aún teniendo en cuenta que hay otras tareas temporizadas, de menor prioridad como la encuesta de la red en diferentes protocolos, el proceso local, la Toma de Estado, etc.

El funcionamiento de la red y reporte de todos los IEDs incluidos en ella se efectuó tal cual lo previsto, sin novedades que destacar, merituando aún más el funcionamiento del sistema operativo.

En el transcurso de los ensayos, se observaron algunos reinicios de la aplicación por activación del Watchdog asociado a alguna de las tareas del esquema Multitask/Multicore. Si bien se habían previsto en éstas los controles correspondientes a Watchdogs y se incluyeron retardos, que ponen en espera a los procesos de mayor prioridad, para ceder *Quantum* de ejecución a otros de menor prioridad, ha habido situaciones donde se evidencia la necesidad de un estudio más profundo de las características y configuración de RTOS para que lo indicado no suceda. Se considera que es más bien un problema de configuración o de incorrecta programación en algún punto, que una falla inherente al SO.

El desempeño del registro de eventos por su parte, ha sido altamente satisfactorio como lo evidencian las pruebas. Queda pendiente el desafío para casos de una cantidad mayor de eventos, donde podrían producirse similares

diferimientos a los ya indicados, que podrían ser inaceptables. Por su parte, el estudio más profundo del caso que se presenta con el geo-posicionamiento de fallas eléctricas descrito más arriba, no se intentó abordarlo en esta etapa debido a que ello implicaría modificaciones importantes al software y la necesidad de contar con GPS con Emisión de Flanco, para determinar la incidencia de la sincronización Externa en el resultado.

A la luz de los resultados globales del proyecto, evaluados durante su desarrollo y las pruebas y en base a las conclusiones alcanzadas, se proponen algunas acciones para el futuro inmediato, tales como:

- Realizar un análisis más profundo y ensayos adicionales sobre la funcionalidad del RTOS, especialmente en lo que respecta a la separación de tareas para múltiples procesadores, dado su desempeño destacado en el trabajo realizado.

- Iniciar una línea de investigación sobre otras necesidades presentes en entornos de Misión Crítica, como los sectores Eléctrico, Químico e Industrial, que podrían beneficiarse de soluciones basadas en microcontroladores de la línea investigada.

- Evaluar la integración de algoritmos para el análisis predictivo y la detección temprana de fallas, mejorando la capacidad del sistema para anticipar y responder a eventos críticos.

Estas acciones podrían fortalecer y expandir los alcances del proyecto, permitiendo abordar nuevos desafíos y aprovechar oportunidades en la aplicación de microcontroladores en entornos críticos y de automatización industrial.

REFERENCIAS

[1] Coulouris G., Dollimore J., Kindberg T., Bair G. “Distributed Systems, Concepts and Design”. Fifth Edition. Addison-Wesley. 2012.

[2] Tanenbaum A., VanSteen V. “Distributed Systems”. Second Edition. Prentice Hall. 2002.

[3] Cristian F., Fetzer C. “The Timed Asynchhronous Distributed System Model” IEEE Transactions on Parallel and Distributed Systems, Vol 10, Nro 6 June 1999.

[4] Fernando G. Tinetti, Ricardo A. López. “Redes de Microcontroladores: Definición, Evaluación y Perspectivas de un Sistema Distribuido". X Workshop de Investigadores en Ciencias de la Computación 2008, Fac. de Ingeniería, Universidad Nacional de La Pampa, Gral. Pico, La Pampa, Argentina, Mayo 5-6 de 2008, pp. 204 – 208.

[5] Ricardo A. López. “Protocolos en Redes de Microcontroladores.” Tesis de Magister: Redes de datos. Director: Fernando F. Tinetti. 1ra Edición - Septiembre 2010. Publicó: Fac. Informática UN La Plata. Volumen 144 hojas.

[6] Ricardo A. López. Nahuel Defossé. "Control de Tránsito con Redes de Microcontroladores". II Congreso Virtual de Microcontroladores y sus Aplicaciones. Título: ISBN: 978-987-25855-5-6. 18 al 29 de Octubre 2010. Universidad Tecnológica Nacional. Entre Rios. Argentina.

[7] Ricardo A. López. Thesis Review: “Protocolos en Redes de Microcontroladores.” (Protocols for Microcontrollers Networks) Special Issue on Research in Computer Science. Vol 11 - No. 1. pags 47-49, April 2011. ISSN 1666-6038. Fac. de Informática. Universidad Nal. de La Plata.

[8] Ricardo A. López, Emilio Pinciroli, Fernando G. Tinetti. " Microcontroladores Asociados a Medición y Comunicaciones en Sistemas SCADA de Energía ". XVI Workshop de Investigadores en Ciencias de la Computación. Ushuaia, 7 y 8 de Mayo de 2014, Universidad de Tierra del Fuego.pp. 646-650.

[9] Fernando G. Tinetti, Ricardo A. López, Sebastián Wahler. "Sincronización de Microcontroladores Interconectados: Evaluación de Factibilidad y Detalles de Implementación". X Workshop de Investigadores en Ciencias de la Computación 2008, Fac. de Ingeniería, Universidad Nacional de La Pampa, Gral. Pico, La Pampa, Argentina, Mayo 5-6 de 2008, pp. 209 – 213.

[10] Fernando G. Tinetti, Ricardo A. López, Marcelo E. Gómez, Sebastián Wahler. "Sincronización de Microcontroladores en Red, Implementación y Evaluación”. XV Congreso Argentino de Ciencias de la Computación 2009, Universidad Nacional de Jujuy, Jujuy, Argentina, Octubre 5 al 9 de 2009, ISBN 978 897 24068-4-1.

[11] Ricardo A. López, Fernando G. Tinetti. " Sincronización Broadcast en Redes Multipunto”. XVII Congreso Argentino de Ciencias de la Computación 2011, Universidad Nacional de La Plata, La Plata, Argentina, Octubre 10 al 14 de 2011, ISBN: 978-950-34-0756-1. JCS&T, Journal of Computer Science & Technology.

[12] Marcelo E. Gómez, Sebastián P. Wahler, Fernando G. Tinetti, Ricardo A. López. " Implementación de Mensajes Rápidos y Valores de Muestreo IEC61850 sobre Ethernet con Microcontroladores". XVI Workshop de Investigadores en Ciencias de la Computación Título: Ushuaia, 7 y 8 de Mayo de 2014, Universidad de Tierra del Fuego.

[13] Ricardo A. López. “Sistema de Semaforización Inteligente”. Revista UNICA de la Asociación de Profesionales Universitarios del Agua y la Energía Eléctrica. (APUAYE) Vol. 107 pags 40-43, Agosto 2009.

[14] Fernando G. Tinetti, Ricardo A. López, Nahuel Defossé. "Microcontroladores DSP y Aplicaciones WEB". XII Workshop de Investigadores en Ciencias de la Computación. 5 y 6 de Mayo de 2010, Calafate, Sta. Cruz. Argentina. pp. 646-650.

[15] Nahuel Defossé, Diego Van Haaster, Lautaro Pecile, Fernando G. Tinetti, Ricardo A. López. " Implementación de Sistemas SCADA Utilizando Lenguajes de Alto Nivel ". XVI Workshop de Investigadores en Ciencias de la Computación. Ushuaia, 7 y 8 de Mayo de 2014, Universidad de Tierra del Fuego.

[16] Defossé N., López R. A., Marcelo E. Gómez, Konstantinoff P., Wahler S., Castro L., Harris G. “Implementación de Middleware Publicador/Subscriptor para Aplicaciones Web de Monitoreo”. WICC 2017 XIX Workshop de Investigadores en Ciencias de la Computación 27 y 28 de Abril’17. Buenos Aires. Libro_WICC_2017 ISBN_978-987-42-5143-5 Pags 181 a 185.